Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Базы Данных (БД)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

«База данных стратегической онлайн-игры»

БГУИР КП 1-40 01 01 104 ПЗ

Студент: гр. 951001 Белых Д.В.

Руководитель: Фадеева Е.Е.

Минск 2022

Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ

Заведующий кафедрой ПОИТ

––––––––––––––––––––––––

(подпись)

––––––––––––––––– 2022 г.

ЗАДАНИЕ

по курсовому проектированию

Студенту     Белых Дмитрию Вячеславовичу –––––––––––––––––––

1. Тема работы   База данных стратегической онлайн-игры

2. Срок сдачи студентом законченной работы––16.12.2022 г.–––

3. Исходные данные к работе   Программа для проектирования базы данных Sparx Systems Enterprise Architect, сервер базы данных Microsoft SQL Server

4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)

Введение.

1.Анализ прототипов, литературных источников и формирование требований к проектируемой базе данных;

2.Анализ предметной области и разработка функциональных требований;

3.Проектирование базы данных;

4.Тестирование, проверка работоспособности и анализ полученных результатов;

Заключение

Список используемой литературы

5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)

1. " База данных стратегической онлайн-игры", инфологическая модель базы данных, А1, плакат.

6. Консультант по курсовой работе

Фадеева Е.Е.

7. Дата выдачи задания 03.09.2022 г.–––––––––––––––––––––––   –

8. Календарный график работы над курсовой работой на весь период проектирования (с обозначением сроков выполнения и процентом от общего объёма работы):

раздел 1 к 15.09.2022 – 15 % готовности работы;

разделы 2, 3 к 15.10.2022 – 30 % готовности работы;

разделы 4, 5 к 15.11.2022 – 60 % готовности работы;

раздел 6 к 15.12.2022 – 90 % готовности работы;

оформление пояснительной записки и графического материала к 14.12.2022 – 100 % готовности работы.

Защита курсовой работы с 13 по 16 декабря 2022 г.–––––––––––––––––––––

РУКОВОДИТЕЛЬ–––––– Е.Е.Фадеева

(подпись)

Задание принял к исполнению –––\_\_\_\_–– Д.В.Белых    03.09.2022 г.

(дата и подпись студента)

## **СОДЕРЖАНИЕ**

СОДЕРЖАНИЕ 4

ВВЕДЕНИЕ………………………………………………………………………..5

1 АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОЙ БАЗЕ ДАННЫХ 7

1.1 Анализ существующих аналогов 6

1.1.1 Видеоигра Clash of Сlans 6

1.1.2 Видеоигра Plants vs Zombies 7

1.1.3 Видеоигра Castle Clash 8

1.2 Постановка задачи 8

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ 9

2.1 Описание функциональности ПС 9

2.2 Выбор СУБД 10

2.4 Выбор типа СУБД 10

3 ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 12

3.1 Проектирование инфологической модели 12

3.2. Нормализация 18

3.3 Создание представлений, функций, процедур и триггеров 19

4 ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ 22

4.1 Тестирование функциональности базы данных 22

ЗАКЛЮЧЕНИЕ 24

СПИСОК ИСТОЧНИКОВ 25

ПРИЛОЖЕНИЕ 1 26

ПРИЛОЖЕНИЕ 2 27

**ВВЕДЕНИЕ**

В качестве темы курсового проекта была выбрана база данных стратегической онлайн-игры.

Два года назад продажи видеоигр, а также игровых устройств и комплектующих достигли рекордных показателей из-за пандемии и всеобщей изоляции. Так, затраты в мире на гейминг увеличились почти на 30% по сравнению с 2019 годом. Между тем, рост популярности компьютерных игр был заметен еще до пандемии и будет сохраняться достаточно долгое время. Более того потребители постепенно откажутся от привычных развлечений в пользу онлайн-интертеймента.

Самыми популярными жанрами игр оказались гонки и шутеры — их указывают игроки со всех платформ. При этом на мобильных устройствах в топ-5 также попадают головоломки, различные вариации настольных игр и симуляторы.

В топ любимых игр так же вошли гонки, спортивные симуляторы, симуляторы жизни, шутеры и стратегии.

Поэтому создание базы данных стратегической онлайн-игры самая важная часть разработки, где необходимо учесть следующие требования:

1. целостность базы данных – требование полноты и непротиворечивости данных;
2. многократное использование данных;
3. быстрый поиск и получение информации по запросам пользователей;
4. простота обновления данных;
5. уменьшение излишней избыточности данных;

# АНАЛИЗ ПРОТОТИПОВ, ЛИТЕРАТУРНЫХ ИСТОЧНИКОВ И ФОРМИРОВАНИЕ ТРЕБОВАНИЙ К ПРОЕКТИРУЕМОй БАЗЕ ДАННЫХ

## Анализ существующих аналогов

Многопользовательская онлайн-игра — это жанр онлайн игр, главной особенностью которого является возможность наслаждаться игрой с сотнями, и даже миллионами других пользователей. Здесь перед вами откроется шанс проявить себя как успешного дипломата, правителя, снайпера, военного тактика, настоящего героя, спасителя целого мира - список многопользовательских игр велик и позволяет найти проект по душе буквально в любой тематике. Такие проекты притягивают геймеров, потому что позволяют играть с друзьями, объединяться с ними в кланы, союзы, альянсы, группировки, да и вообще помогают найти новых знакомых по всему миру.

### **1.1.1 Видеоигра Clash of Сlans**

Clash of Сlans — стратегическая игра, созданная финской студией-разработчиком Supercell для мобильных устройств. Распространяется по модели free-to-play. Игроку нужно тщательно планировать и рассчитывать каждое действие, поскольку игроку придется не только оборонять свою деревню от нападений других пользователей, но и атаковать их самому



Рисунок 1.1 – Изображение из игры Clash of Сlans

Плюсы:

* Есть возможность участвовать в клановых битвах;
* Простота в управлении;
* Завлекает на много часов;

Минусы:

* Невозможно управлять воинами во время нападения;
* Достаточно высокие цены на донат;

### **1.1.2 Видеоигра Plants vs Zombies**

Plants vs Zombies расскажут вам историю противостояние природы и нежити. В вашем распоряжении есть небольшой участок земли и горстка семян, из которых вам предстоит вырастить растения, способные уничтожать зомби. По мере прохождения вы будете открывать и покупать все новые и новые семена. Всего их существует более 40 штук. Разделить их можно на 4 типа: поддержка, защита, ближний или дальний бой. Основная ваша задача — не пропустить зомби дальше вашей фермы.



Рисунок 1.2 – Изображение из игры Plants vs Zombies

Плюсы:

* Простота в управлении;
* Увлекательный сюжет;
* Разнообразие уровней;
* Бесплатная

Минусы:

* Отсутствие русского языка при установке;

## 1.1.3 Видеоигра Castle Clash

Castle Clash – это игра стратегия с элементами менеджмента, в которой вам понадобится накапливать ресурсы, строить здания, чтобы создать деревню, нанимать армию и создавать множество различных созданий. Рисунок 1.3 – Деревня одного из игроков игры Castle Clash

Плюсы:

* Простота в управлении и понимании;
* Бесплатная;

Минусы:

* Новые обновления делают игру сложной для понимания;
* Нет русскоязычной локализации.

## Постановка задачи

В настоящее время существует масса многопользовательских стратегических онлайн-игр с богатой функциональностью, однако мало из них подходят для новичка, т.к. большинство из них сложны для понимания. Поэтому данная курсовая работа направлена на разработку базы данных стратегической онлайн-игры, которая будет содержать только основы игры в жанре стратегия.

К основным целям данной курсовой работы можно отнести:

* изучить предметную область;
* проанализировать логическую и физическую модель представления данных;
* создать базу данных;
* дополнить ее необходимым функционалом.

# АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И РАЗРАБОТКА ФУНКЦИОНАЛЬНЫХ ТРЕБОВАНИЙ

## 2.1 Описание функциональности ПС

В результате анализа требований к программному средству были сформированы следующие функциональные требования:

* отображение аккаунта игрока;
* отображение клана игрока;
* отображение построек игрока;
* отображение исследований игрока;
* отображение юнитов игрока;
* отображение истории посещений игрока;
* возможность просмотра ресурсов на карте;

В результате анализа требований к ПС база данных должна иметь все необходимые таблицы для реализации функциональных требований, а также следует:

* дополнить ее функциональность триггерами;
* дополнить ее функциональность процедурами;
* дополнить ее функциональность функциями;
* дополнить ее функциональность представлениями.

Программное средство может представлять собой программное средство, которое игрок сможет использовать для участия в онлайн-игре.

Клиентская часть приложения может быть реализована как программное средство. На данный момент – это самый удобный способ доступа для взаимодействия с игрой и другими игроками.

Серверная часть приложения может быть реализована как набор микросервисов, взаимодействующих между собой по стандартизированному API, а также общей базе данных.

База данных должна быть реализована в соответствии со всеми требованиями к программному средству, а также быть нормализована минимум до третей нормальной формы.

## 2.2 Выбор СУБД

Таблица 2.1 – Сравнение СУБД

|  |  |  |
| --- | --- | --- |
| **Тип СУБД** | **Когда выбирать** | **Примеры популярных СУБД** |
| Реляционные | Нужна транзакционность; высокая нормализация; большая доля операций на вставку | Oracle, MySQL, Microsoft SQL Server, PostgreSQL |
| Ключ-значение | Задачи кэширования и брокеры сообщений | Redis, Memcached |
| Документные | Для хранения объектов в одной сущности, но с разной структурой; хранение структур на основе JSON | CouchDB, MongoDB, Amazon DocumentDB |
| Графовые | Задачи подобные социальным сетям; системы оценок и рекомендаций | Neo4j, Amazon Neptune, InfiniteGraph, InfoGrid |
| Колоночные | Хранилища данных; выборки со сложными аналитическими вычислениями; количество строк в таблице превышает сотни миллионов | Vertica, ClickHouse, Google BigTable, Sybase \ SAP IQ, InfoBright, Cassandra |

Реляционные СУБД чаще всего используются для построения решений OLTP (Online Transaction Processing). В таких решениях СУБД работает с небольшими по размерам транзакциями, но идущими большим потоком, и при этом от системы требуется минимальное время отклика, а также возможность, при определенных условиях, отменить любые изменения выполняемых в рамках транзакции. Т.к. мы строим систему, в рамках которой требуется хранить значительное количество сущностей (таблиц), с различными типами связей между ними (один-к-одному, один-к-многим, многие-ко-многим), то наш выбор – реляционные СУБД.

## 2.3 Выбор типа СУБД

Чтобы упростить работу с хранилищами данных и повысить эффективность их применения, создаются специализированные системы управления. Одной из наиболее популярных является разработка от Microsoft – SQL Server. Первый релиз платформы опубликован еще в 1989 году, а последняя версия выпущена в 2019 году (проект продолжает развиваться).

Преимущества решения:

- Тесная интеграция с операционной системой Windows.

- Высокая производительность, отказоустойчивость.

- Поддержка многопользовательской среды.

- Расширенные функции резервирования данных.

- Работа с удаленным подключением.

Каждый выпуск включает в себя несколько специализированных редакций. Это снижает сложность внедрения и затраты на процесс разработки собственных решений, адаптированных для «узких» задач. При написании программного кода активно используется интеграция с продуктами Microsoft, например, с платформой Visual Studio.

Прямые конкуренты на рынке – Oracle Database, PostgreSQL. Первый проект коммерческий, он создан для поддержки крупных компаний, поэтому сопоставим по возможностям с MS SQL Server. Второй же распространяется на бесплатной основе и не «блещет» функциональностью, хотя весьма популярен среди многих разработчиков (аналог от Oracle MySQL).

# ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 3.1 Проектирование инфологической модели

Таблица 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Cущность | Атрибут | Описание Атрибута |
| Сlan – таблица об объединении игроков в однин клан | id | идентификатор клана, первичный ключ |
| clan\_name | Уникальное название клана |
| date\_of\_founded | Дата создания |
| date\_of\_disbanded | Дата расформирования |
| Clan\_member– таблица участников клана | member\_id | Уникальный идентефикатор участника, первичный ключ |
| user\_id | Идентефикатор игрока, внешний ключ |
| clan\_id | Идентефикатор клана, внешний ключ |
| date\_from | Дата вступления |
| date\_for | Дата ухода |
| clan\_role\_id | идентефикатор роли в клане, внешний ключ |
| Member\_history – таблица длительности ролей игроков в клане | id | Уникальный идентификатор истории , первичный ключ |
| clan\_member\_id | Идентефикатор участника, внешний ключ |
| clan\_role\_id | Идентефикатор роли, внешний ключ |
| date\_from | Дата получения роли в клане |
| date\_for | Дата окончания роли в клане |
| Clan\_role – таблица ролей в клане | role\_id | Уникальный идентификатор роли , первичный ключ |
| name | Назание роли |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Role\_permissions – Таблица для соотношения ролей и действий | member\_role\_id | Идентификатор пользователя внешний и первичный ключ |
| member\_permissions\_id | Идентификатор пользователя внешний и первичный ключ |
| Member\_opportunity – Таблица действий в клане | op\_id | Уникальный идентификатор права, первичный ключ |
| name | Название права |
| User\_account – Таблица списка всех зарегестрированных игроков | user\_id | Уникальный идентификатор игрока, первичный ключ |
| first\_name | Имя |
| last\_name | Фамилия |
| nickname | Никнейм |
| password | Пароль |
| gender | Пол |
| user\_role\_id | Идентефикатор роли, внешний ключ |
| email | Электронная почта |
| User\_account\_history – Таблица истории входа в игру | id | Уникальный идентификатор входа, первичный ключ |
| user\_account\_id | Идентификатор игрока, внешний ключ |
| signin\_time | Время входа |
| signout\_time | Время выхода |
| device\_info | Устройство входа |
| ip\_address | Адрес |
| User\_roles – Таблица ролей игроков | role\_id | Уникальный идентификатор роли, первичный ключ |
| name | Название роли |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| User\_permissions – Таблица прав | perm\_id | Уникальный идентификатор, первичный ключ |
| user\_role\_id | Идентификатор роли, внешний ключ |
| user\_permissions\_id | Идентификатор действия, внешний ключ |
| Group\_movement – Таблица для описания передвижений группы юнитов | id | Уникальный идентификатор, первичный ключ |
| user\_id | Идентификатор игрока, внешний ключ |
| location\_from | Из какой локации |
| location\_to | В какую локацию |
| arrival\_time | Время прибытия |
| return\_time | Время возвращения |
| Group\_members – Таблица участников группы | id | Уникальный идентификатор, первичный ключ |
| count | Количество юнитов |
| unit\_id | Идентификатор юнита, внешний ключ |
| group\_move\_id | Идентификатор перемещения, внешний ключ |
| Units\_on\_location – Таблица готовых юнитов на карте | id | Уникальный идентификатор, первичный ключ |
| unit\_id | Идентификатор юнита, внешний ключ |
| location\_id | Идентификатор локации, внешний ключ |
| Count | Количество юнитов |
| Units – таблица всех юнитов | id | Уникальный идентификатор юнита, первичный ключ |
| name | Название юнита |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Unit\_characteristic – Таблица характеристик каждого юнита | id | Уникальный идентификатор характеристики, первичный ключ |
| unit\_id | Идентификатор юнита, внешний ключ |
| name | Название характеристики |
| value | Уровень характеристики |
| Resources\_in\_group – Таблица ресурсов, которые принесет группа | id | Уникальный идентификатор, первичный ключ |
| resource\_id | Идентификатор ресурса, внешний ключ |
| group\_move\_id | Идентификатор перемещения, внешний ключ |
| count | Количество |
| Unit\_cost – Таблица стоимости юнитов | id | Уникальный идентификатор, первичный ключ |
| unit\_id | Идентификатор юнита, внешний ключ |
| resource\_id | Идентификатор ресурса, внешний ключ |
| cost | Стоимость |
| Resource – Таблица ресурсов | id | Уникальный идентификатор, первичный ключ |
| name | Название |
| Research\_unit – Таблица исследовательских работ, нужных для производства данного типа юнитов | id | Уникальный идентификатор, первичный ключ |
| research\_id | Идентификатор исследования, внешний ключ |
| unit\_id | Идентификатор юнита, внешний ключ |
| level\_required | Необходимый уровень |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Resources\_on\_locations – Таблица ресурсов в локации | id | Уникальный идентификатор ресурса, первичный ключ |
| resource\_id | Идентификатор ресурса, внешний ключ |
| location\_id | Идентификатор локации, внешний ключ |
| count | Количество |
| Location – Таблица локаций | location\_id | Уникальный идентификатор локации, первичный ключ |
| name | Название |
| user\_id | Идентификатор игрока, внешний ключ |
| Location\_build – Таблица построек в локации | id\_ | Уникальный идентификатор построек, первичный ключ |
| is\_upgrade | Флаг, происходит ли сейчас улучшение или нет |
| level | Уровень постройки |
| structure\_id | Идентификатор структуры, внешний ключ |
| location\_id | Идентификатор локации, внешний ключ |
| upgrate\_time | Время завершения улучшения |
| Structure – Таблица всех структур | str\_id | Уникальный идентификатор структуры, первичный ключ |
| name | Название постройки |
| built\_time | Время строительства |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Structure\_upgrade – Таблица улучшений построек | id | Уникальный идентификатор, первичный ключ |
| structure\_id | Идентификатор структуры, внешний ключ |
| resource\_id | Идентификатор ресурса, внешний ключ |
| res\_for\_upgrade | Необходимые ресурсы для улучшения |
| production\_res | Количество ресурсов которое будет производиться после улучшения |
| Structure\_required –  Таблица для улучшения структуры | id | Уникальный идентификатор, первичный ключ |
| structure\_id | Идентификатор структуры которую надо улучшить, внешний ключ |
| structure\_required\_id | Идентификатор структуры, которая должна быть построена, внешний ключ |
| level | Уровень необоходимого здания, необходимый, чтобы улучшить другое здание |
| Necessary\_research –  Таблица необходимых исследований, чтобы построить структуру | id | Уникальный идентификатор, первичный ключ |
| research\_id | Идентификатор исследования, внешний ключ |
| structure\_id | Идентификатор структуры, внешний ключ |
| level\_required | Необходимый уровень исследований |

Продолжение таблицы 3.1 – Инфологическая модель

|  |  |  |
| --- | --- | --- |
| Research –  Таблица всех исследований | id | Уникальный идентификатор, первичный ключ |
| name | Название исследования |
| research\_time | Время исследования |
| Research\_requared – Таблица необходимых исследований | id | Уникальный идентификатор, первичный ключ |
| research\_id | Идентификатор исследования которое надо улучшить, внешний ключ |
| research\_required\_id | Идентификатор исследования, которое должно быть изучено, внешний ключ |
| level | Необохдимый уровень исследования |
| Research\_level– Таблица исследований для каждого игрока | id | Уникальный идентификатор, первичный ключ |
| research\_id | Идентификатор исследования, внешний ключ |
| current\_level | Текущий уровень |
| is\_upgrade | Улучшается ли исследование сейчас |
| upgrade\_time | Время конца исследования |
| user\_id | Идентификатор пользователя, внешний ключ |

## Нормализация

Проверка на первую нормальную форму. Просмотрев отношения не было замечено сложных и многозначные атрибутов. Таблицы приведены к первой нормальной форме.

Проверка на вторую нормальную форму. Просмотрев отношения, описанные выше, можно сказать, что не ключевые атрибуты этих отношений функционально полно зависят от первичных ключей. Следовательно, отношения приведены ко второй нормальной форме.

Проверка на третью нормальную форму. Транзитивные функциональные зависимости отсутствуют, поэтому отношения соответствуют третьей нормальной форме.

## 3.3 Создание представлений, функций, процедур и триггеров

В процессе проектирование базы данных большое внимание было уделено сбору статистики футбольных матчей. Поэтому был создан набор представлений, функций, процедур и триггеров для удобного доступа к данным.

Таблица 3.2 – Проектируемые вспомогательные элементы

|  |  |  |
| --- | --- | --- |
| Сущность | Тип | Назначение |
| User\_account | Представление users\_research | Показывает всех игроков, какие у них исследованы умения и их уровень |
| Любая сущность из базы данных | Процедура SHOW\_FOREIGN\_KEY\_AND\_TABLE | Показывает внешние ключи, которые имеет указанная таблица |
| User\_account\_history | Триггер add\_to\_user\_acc\_history | При добавлении нового игрока, записывает его игровую сессию в таблицу истории |
| Matches | Фунция GET\_user\_account | Позволяет получить конкретную группу заданного турнира |
| Matches | Фунция GET\_disbanded\_clans | Позволяет получить матчи, где забито больше голов, чем указано |
| Players | Фунция GET\_count\_of\_same\_structures\_on\_location | Получает игроков, которые моложе, чем заданный возраст |

Листинг представлений, функций, процедур и триггеров:

USE [Video Game]

GO

CREATE OR ALTER VIEW [users\_research] AS

SELECT [nickname] as [Player],[name] as [Skill],[current\_level] as [level]

FROM [User\_account]

JOIN [Research\_level]

ON [dbo].[User\_account].[user\_id]=[dbo].[Research\_level].[user\_id]

JOIN [Research]

ON [dbo].[Research\_level].[user\_id]=[dbo].[Research].[id]

GO

SELECT \* FROM [dbo].[users\_research]

GO

CREATE OR ALTER PROCEDURE SHOW\_FOREIGN\_KEY\_AND\_TABLE

@table\_name NVARCHAR(150)

WITH EXECUTE AS OWNER

AS

DECLARE @query\_text NVARCHAR(1000) = '';

SET @query\_text = 'SELECT [object\_type], [object\_name], [sys].[tables].[name] as [foreign\_table] FROM (SELECT ''foreign\_key'' AS [object\_type],

[constraint\_name] AS [object\_name]

FROM [information\_schema].[table\_constraints]

WHERE [table\_catalog] = DB\_NAME()

AND [table\_name] = ''\_FP\_TABLE\_NAME\_PLACEHOLDER\_''

AND [constraint\_type] = ''FOREIGN KEY'') as table\_foreing\_keys

JOIN [sys].[foreign\_keys]

ON [name] = [object\_name]

JOIN [sys].[tables]

ON [sys].[tables].[object\_id] = [referenced\_object\_id]';

SET @query\_text = REPLACE(@query\_text, '\_FP\_TABLE\_NAME\_PLACEHOLDER\_',

@table\_name);

EXECUTE sp\_executesql @query\_text;

GO

EXECUTE SHOW\_FOREIGN\_KEY\_AND\_TABLE 'User\_account';

GO

CREATE OR ALTER TRIGGER [add\_to\_user\_acc\_history]

ON [dbo].[User\_account]

AFTER INSERT

AS

DECLARE @inserter\_user\_id int

DECLARE @new\_signin\_time bigint

DECLARE @new\_signout\_time bigint

DECLARE @device\_info varchar(50)

DECLARE @ip\_address varchar(50)

SET @new\_signin\_time = 21323

SET @new\_signout\_time = NULL

SET @device\_info = 'phone'

SET @ip\_address = '23.14.53.23.54'

SELECT @inserter\_user\_id = (SELECT [user\_id] FROM inserted)

INSERT INTO [User\_account\_history]

VALUES (@inserter\_user\_id,@new\_signin\_time,@new\_signout\_time,@device\_info,@ip\_address)

GO

SET IDENTITY\_INSERT [dbo].[User\_account] ON

INSERT INTO [dbo].[User\_account] ([user\_id],[first\_name],[last\_name],[nickname] ,[password],[gender],[user\_role\_id],[email])

VALUES (7,'Trigger','Mihael','Misha','123445','m',4,'GFG@mail.ru')

SET IDENTITY\_INSERT [dbo].[User\_account] OFF

GO

CREATE OR ALTER FUNCTION [dbo].[GET\_user\_account](@user\_nickname VARCHAR(50))

RETURNS TABLE

AS

RETURN (SELECT \* FROM [dbo].[User\_account] WHERE [nickname] = @user\_nickname);

GO

SELECT \* FROM [dbo].[GET\_user\_account] ('Awik')

GO

CREATE OR ALTER FUNCTION [dbo].[GET\_disbanded\_clans]()

RETURNS TABLE

AS

RETURN (SELECT \* FROM [dbo].[Clan] WHERE [date\_of\_disbanded] IS NOT NULL);

GO

SELECT \* FROM [dbo].[GET\_disbanded\_clans] ()

GO

CREATE OR ALTER FUNCTION [dbo].[GET\_count\_of\_same\_structures\_on\_location](@Nes\_structure\_id int,@location\_id int)

RETURNS TABLE

AS

RETURN (SELECT COUNT([is\_upgrade]) as [count\_of\_same] FROM [dbo].[Location\_build] WHERE @location\_id=[location\_id] AND @Nes\_structure\_id=[structure\_id] );

GO

SELECT \* FROM [dbo].[GET\_count\_of\_same\_structures\_on\_location] (1,1)

GO

# ТЕСТИРОВАНИЕ, ПРОВЕРКА РАБОТОСПОСОБНОСТИ И АНАЛИЗ ПОЛУЧЕННЫХ РЕЗУЛЬТАТОВ

Проведено тестирование базы данных на персональном компьютере с установленной операционной системой Windows 10.

С помощью инструментов среды разработки Enterprise Architect был произведён экспорт полученного SQL скрипта для создание БД. Данный скрипт был запущен на сервере Microsoft SQL Server и завершился успешно. Это свидетельствует о корректном синтаксе полученного скрипта.

## Тестирование функциональности базы данных

Таблица 4.1 – тестирование функциональности программного средства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Номер**  **теста** | **Тестируемая функциональность** | **Последовательность действий** | **Ожидаемый результат** | **Полученный результат** |
| 0 | Вставка данных | 1. Создать и запустить скрипт по созданию таблиц в базе данных.  2. Запуск скрипта | В базу данных были добавлены все таблицы | Тест пройден |
| 1 | 1. Создать и запустить скрипт по наполнении таблиц данными.  2. Запуск скрипта | В базу данных были добавленны данные | Тест пройден |
| 2 | Чтение данных | 1. Создать запрос по чтению данных из таблицы игроков  2. Запустить запрос | Заполненная таблица игроков | Тест пройден |
| 3 | 1. Создать запрос по чтению данных из таблицы ресурсы  2. Запустить запрос | Заполненная таблица ресурсов | Тест пройден |
| 4 | 1. Создать запрос по чтению данных из таблицы роли  2. Запустить запрос | Заполненная таблица ролей | Тест пройден |

Таблица 4.1 – тестирование функциональности программного средства

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 6 | Тестирование представления | 1.Создать представление  2. Запустить преставление | Корректная работа представления | Тест пройден |
| 7 | Тестирование функций | 1.Создать функцию по поиску конкретного игрока и вывод его профиля  2. Выполнить функцию | Корректная работа функции на входные данные | Тест пройден |
| 8 | 1.Создать функцию по поиску кланов, которые расформированы  2. Выполнить функцию | Корректная работа функции на входные данные. | Тест пройден |
| 9 | 1.Создать функцию для просмотка на конкретной локации сколько одинаковых построек  2. Выполнить функцию | Корректная работа функции на входные данные. | Тест пройден |
| 10 | Тестирование процедур | 1.Создать процедуру для отображения всех внешних ключей таблицы  2. Выполнить процедуру | Корректная работа процедуры на входные данные. | Тест пройден |
| 11 | Тестирование триггера | 1.Создать триггер при  добавлении нового игрока, записывает его игровую сессию в таблицу истории  2. Выполнить триггер | Корректная работа триггера на входные данные | Тест пройден |

# ЗАКЛЮЧЕНИЕ

В ходе написания данной курсовой работы были изучены и усовершенствованы принципы проектирования баз данных:

* нормализация;
* построение связей;
* мощности связей;
* использование соответствующих типов данных.

Изучена среда проектирования баз данных Enterprise Architect, которая использовалась для проектированя, а также Microsoft SQL Server для проверки работы скриптов по созданию базы данных и выполнения запросов к ней.

Был получен опыт построения запросов на языке SQL для управления данными и их выборки.

В результате была получена база данных под конкретную предметную область, которую можно использовать в дальнейшей разработке приложения.

# СПИСОК ИСТОЧНИКОВ

[1] Реляционные базы данных в примерах: практическое пособие для программистов и тестировщиков / С. С. Куликов. — Минск: Четыре четверти, 2020. — 424 с.

[2] Работа с MySQL, MS SQL Server и Oracle в примерах: практ. пособие / С. С. Куликов. — 2-е изд. — Минск: Четыре четверти, 2021. — 600 с

[3] Wikipedia [Электронный ресурс] – Электронные данные – Режим доступа: <https://ru.wikipedia.org/wiki/Microsoft_SQL_Server>

[5] Карпова Т.С., Базы данных. Модели, разработка, реализация. СПб.: Питер, 2002. – 304 с.

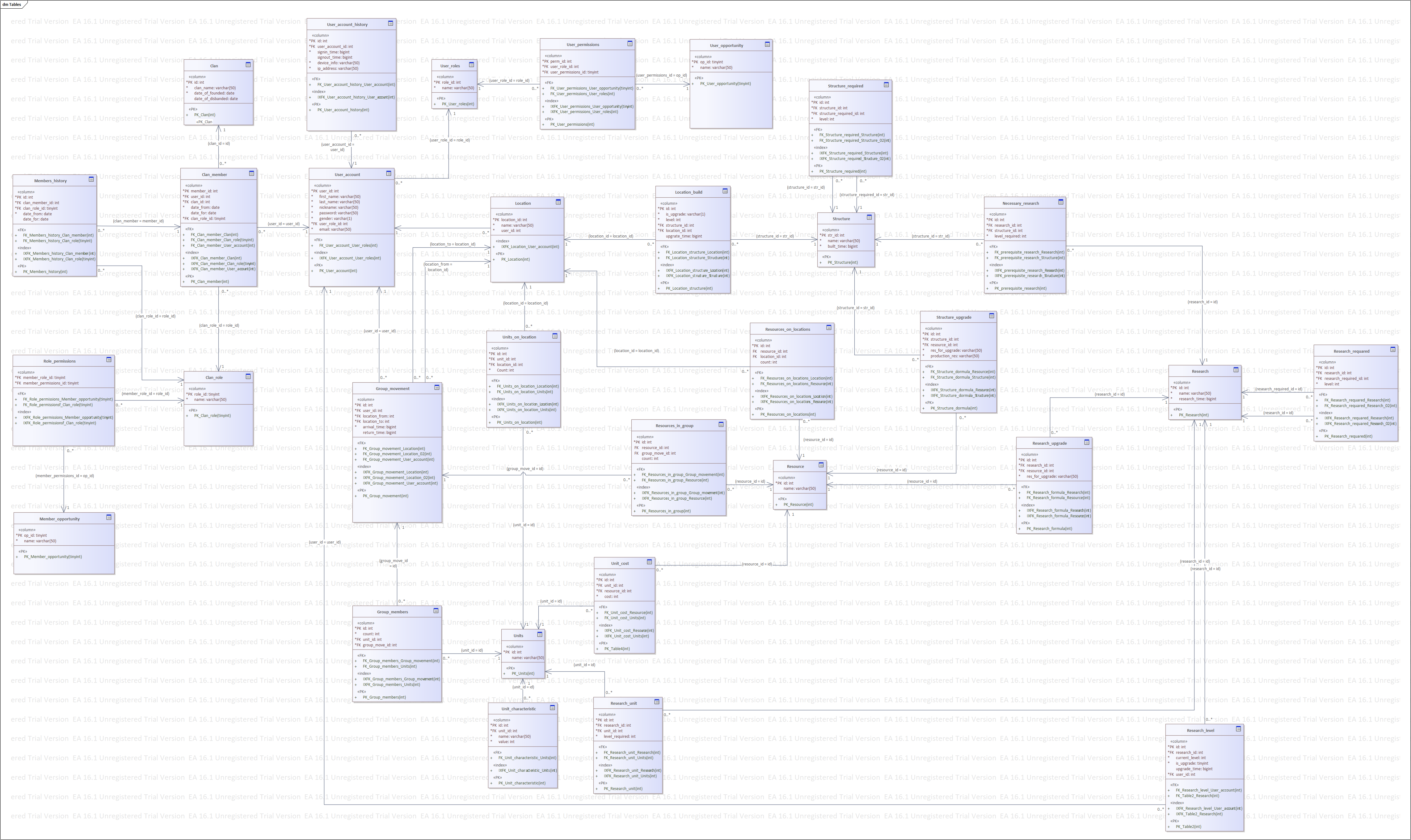
[6] Сайт www.squarespace.com [Электронный ресурс] – Электронные данные – Режим доступа: www.squarespace.com

[7] Сайт w3schools.com [Электронный ресурс] – Электронные данные – Режим доступа: https://www.w3schools.com/sql/

[8] Сайт metanit.com [Электронный ресурс] – Электронные данные – Режим доступа: https://metanit.com/sql/sqlserver/

# ПРИЛОЖЕНИЕ 1

**Схема базы данных**



# ПРИЛОЖЕНИЕ 2

**Скрипт для генерации базы данных**

/\* ---------------------------------------------------- \*/

/\* Generated by Enterprise Architect Version 16.1 \*/

/\* Created On : 16-дек.-2022 04:58:07 \*/

/\* DBMS : SQL Server 2012 \*/

/\* ---------------------------------------------------- \*/

/\* Drop Foreign Key Constraints \*/

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Clan\_member\_Clan]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Clan\_member] DROP CONSTRAINT [FK\_Clan\_member\_Clan]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Clan\_member\_Clan\_role]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Clan\_member] DROP CONSTRAINT [FK\_Clan\_member\_Clan\_role]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Clan\_member\_User\_account]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Clan\_member] DROP CONSTRAINT [FK\_Clan\_member\_User\_account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Group\_members\_Group\_movement]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Group\_members] DROP CONSTRAINT [FK\_Group\_members\_Group\_movement]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Group\_members\_Units]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Group\_members] DROP CONSTRAINT [FK\_Group\_members\_Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Group\_movement\_Location]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Group\_movement] DROP CONSTRAINT [FK\_Group\_movement\_Location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Group\_movement\_Location\_02]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Group\_movement] DROP CONSTRAINT [FK\_Group\_movement\_Location\_02]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Group\_movement\_User\_account]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Group\_movement] DROP CONSTRAINT [FK\_Group\_movement\_User\_account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Location\_structure\_Location]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Location\_build] DROP CONSTRAINT [FK\_Location\_structure\_Location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Location\_structure\_Structure]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Location\_build] DROP CONSTRAINT [FK\_Location\_structure\_Structure]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Member\_history\_Clan\_member]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Member\_history] DROP CONSTRAINT [FK\_Member\_history\_Clan\_member]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_prerequisite\_research\_Research]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Necessary\_research] DROP CONSTRAINT [FK\_prerequisite\_research\_Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_prerequisite\_research\_Structure]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Necessary\_research] DROP CONSTRAINT [FK\_prerequisite\_research\_Structure]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_level\_User\_account]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_level] DROP CONSTRAINT [FK\_Research\_level\_User\_account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Table2\_Research]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_level] DROP CONSTRAINT [FK\_Table2\_Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_requared\_Research]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_requared] DROP CONSTRAINT [FK\_Research\_requared\_Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_requared\_Research\_02]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_requared] DROP CONSTRAINT [FK\_Research\_requared\_Research\_02]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_unit\_Research]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_unit] DROP CONSTRAINT [FK\_Research\_unit\_Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_unit\_Units]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_unit] DROP CONSTRAINT [FK\_Research\_unit\_Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_formula\_Research]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_upgrade] DROP CONSTRAINT [FK\_Research\_formula\_Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Research\_formula\_Resource]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Research\_upgrade] DROP CONSTRAINT [FK\_Research\_formula\_Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Resources\_in\_group\_Group\_movement]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Resources\_in\_group] DROP CONSTRAINT [FK\_Resources\_in\_group\_Group\_movement]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Resources\_in\_group\_Resource]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Resources\_in\_group] DROP CONSTRAINT [FK\_Resources\_in\_group\_Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Resources\_on\_locations\_Location]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Resources\_on\_locations] DROP CONSTRAINT [FK\_Resources\_on\_locations\_Location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Resources\_on\_locations\_Resource]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Resources\_on\_locations] DROP CONSTRAINT [FK\_Resources\_on\_locations\_Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Role\_permissions\_Member\_opportunity]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Role\_permissions] DROP CONSTRAINT [FK\_Role\_permissions\_Member\_opportunity]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Role\_permissionsf\_Clan\_role]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Role\_permissions] DROP CONSTRAINT [FK\_Role\_permissionsf\_Clan\_role]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Structure\_required\_Structure]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Structure\_required] DROP CONSTRAINT [FK\_Structure\_required\_Structure]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Structure\_required\_Structure\_02]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Structure\_required] DROP CONSTRAINT [FK\_Structure\_required\_Structure\_02]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Structure\_dormula\_Resource]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Structure\_upgrade] DROP CONSTRAINT [FK\_Structure\_dormula\_Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Structure\_dormula\_Structure]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Structure\_upgrade] DROP CONSTRAINT [FK\_Structure\_dormula\_Structure]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Unit\_characteristic\_Units]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Unit\_characteristic] DROP CONSTRAINT [FK\_Unit\_characteristic\_Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Unit\_cost\_Resource]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Unit\_cost] DROP CONSTRAINT [FK\_Unit\_cost\_Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Unit\_cost\_Units]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Unit\_cost] DROP CONSTRAINT [FK\_Unit\_cost\_Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Units\_on\_location\_Location]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Units\_on\_location] DROP CONSTRAINT [FK\_Units\_on\_location\_Location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_Units\_on\_location\_Units]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [Units\_on\_location] DROP CONSTRAINT [FK\_Units\_on\_location\_Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_User\_account\_User\_roles]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [User\_account] DROP CONSTRAINT [FK\_User\_account\_User\_roles]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_User\_account\_history\_User\_account]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [User\_account\_history] DROP CONSTRAINT [FK\_User\_account\_history\_User\_account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_User\_permissions\_User\_opportunity]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [User\_permissions] DROP CONSTRAINT [FK\_User\_permissions\_User\_opportunity]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[FK\_User\_permissions\_User\_roles]') AND OBJECTPROPERTY(id, N'IsForeignKey') = 1)

ALTER TABLE [User\_permissions] DROP CONSTRAINT [FK\_User\_permissions\_User\_roles]

GO

/\* Drop Tables \*/

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Clan]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Clan]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Clan\_member]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Clan\_member]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Clan\_role]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Clan\_role]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Group\_members]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Group\_members]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Group\_movement]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Group\_movement]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Location]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Location\_build]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Location\_build]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Member\_history]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Member\_history]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Member\_opportunity]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Member\_opportunity]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Necessary\_research]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Necessary\_research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Research]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Research]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Research\_level]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Research\_level]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Research\_requared]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Research\_requared]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Research\_unit]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Research\_unit]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Research\_upgrade]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Research\_upgrade]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Resource]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Resource]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Resources\_in\_group]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Resources\_in\_group]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Resources\_on\_locations]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Resources\_on\_locations]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Role\_permissions]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Role\_permissions]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Structure]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Structure]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Structure\_required]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Structure\_required]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Structure\_upgrade]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Structure\_upgrade]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Unit\_characteristic]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Unit\_characteristic]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Unit\_cost]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Unit\_cost]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Units]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Units]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[Units\_on\_location]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [Units\_on\_location]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[User\_account]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [User\_account]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[User\_account\_history]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [User\_account\_history]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[User\_opportunity]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [User\_opportunity]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[User\_permissions]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [User\_permissions]

GO

IF EXISTS (SELECT 1 FROM dbo.sysobjects WHERE id = object\_id(N'[User\_roles]') AND OBJECTPROPERTY(id, N'IsUserTable') = 1)

DROP TABLE [User\_roles]

GO

/\* Create Tables \*/

CREATE TABLE [Clan]

(

[id] int NOT NULL IDENTITY (1, 1),

[clan\_name] varchar(50) NOT NULL,

[date\_of\_founded] date NOT NULL,

[date\_of\_disbanded] date NULL

)

GO

CREATE TABLE [Clan\_member]

(

[member\_id] int NOT NULL IDENTITY (1, 1),

[user\_id] int NOT NULL,

[clan\_id] int NOT NULL,

[date\_from] date NOT NULL,

[date\_for] date NULL,

[clan\_role\_id] tinyint NOT NULL

)

GO

CREATE TABLE [Clan\_role]

(

[role\_id] tinyint NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL

)

GO

CREATE TABLE [Group\_members]

(

[id] int NOT NULL IDENTITY (1, 1),

[count] int NOT NULL,

[unit\_id] int NOT NULL,

[group\_move\_id] int NOT NULL

)

GO

CREATE TABLE [Group\_movement]

(

[id] int NOT NULL IDENTITY (1, 1),

[user\_id] int NOT NULL,

[location\_from] int NOT NULL,

[location\_to] int NOT NULL,

[arrival\_time] bigint NOT NULL,

[return\_time] bigint NULL

)

GO

CREATE TABLE [Location]

(

[location\_id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL,

[user\_id] int NOT NULL

)

GO

CREATE TABLE [Location\_build]

(

[id] int NOT NULL IDENTITY (1, 1),

[is\_upgrade] varchar(1) NOT NULL,

[level] int NOT NULL,

[structure\_id] int NOT NULL,

[location\_id] int NOT NULL,

[upgrate\_time] bigint NULL

)

GO

CREATE TABLE [Member\_history]

(

[id] int NOT NULL IDENTITY (1, 1),

[clan\_member\_id] int NOT NULL,

[clan\_role\_id] tinyint NOT NULL,

[date\_from] date NOT NULL,

[date\_for] date NULL

)

GO

CREATE TABLE [Member\_opportunity]

(

[op\_id] tinyint NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL

)

GO

CREATE TABLE [Necessary\_research]

(

[id] int NOT NULL IDENTITY (1, 1),

[research\_id] int NOT NULL,

[structure\_id] int NOT NULL,

[level\_required] int NOT NULL

)

GO

CREATE TABLE [Research]

(

[id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL,

[research\_time] bigint NOT NULL

)

GO

CREATE TABLE [Research\_level]

(

[id] int NOT NULL IDENTITY (1, 1),

[research\_id] int NOT NULL,

[current\_level] int NOT NULL,

[is\_upgrade] tinyint NOT NULL,

[upgrade\_time] bigint NULL,

[user\_id] int NOT NULL

)

GO

CREATE TABLE [Research\_requared]

(

[id] int NOT NULL IDENTITY (1, 1),

[research\_id] int NOT NULL,

[research\_required\_id] int NOT NULL,

[level] int NOT NULL

)

GO

CREATE TABLE [Research\_unit]

(

[id] int NOT NULL IDENTITY (1, 1),

[research\_id] int NOT NULL,

[unit\_id] int NOT NULL,

[level\_required] int NOT NULL

)

GO

CREATE TABLE [Research\_upgrade]

(

[id] int NOT NULL IDENTITY (1, 1),

[research\_id] int NOT NULL,

[resource\_id] int NOT NULL,

[res\_for\_upgrade] varchar(50) NOT NULL

)

GO

CREATE TABLE [Resource]

(

[id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NULL

)

GO

CREATE TABLE [Resources\_in\_group]

(

[id] int NOT NULL IDENTITY (1, 1),

[resource\_id] int NULL,

[group\_move\_id] int NULL,

[count] int NULL

)

GO

CREATE TABLE [Resources\_on\_locations]

(

[id] int NOT NULL IDENTITY (1, 1),

[resource\_id] int NULL,

[location\_id] int NULL,

[count] int NULL

)

GO

CREATE TABLE [Role\_permissions]

(

[member\_role\_id] tinyint NOT NULL,

[member\_permissions\_id] tinyint NOT NULL

)

GO

CREATE TABLE [Structure]

(

[str\_id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL,

[built\_time] bigint NOT NULL

)

GO

CREATE TABLE [Structure\_required]

(

[id] int NOT NULL IDENTITY (1, 1),

[structure\_id] int NOT NULL,

[structure\_required\_id] int NOT NULL,

[level] int NOT NULL

)

GO

CREATE TABLE [Structure\_upgrade]

(

[id] int NOT NULL IDENTITY (1, 1),

[structure\_id] int NOT NULL,

[resource\_id] int NOT NULL,

[res\_for\_upgrade] varchar(50) NOT NULL,

[production\_res] varchar(50) NOT NULL

)

GO

CREATE TABLE [Unit\_characteristic]

(

[id] int NOT NULL IDENTITY (1, 1),

[unit\_id] int NOT NULL,

[name] varchar(50) NOT NULL,

[value] int NOT NULL

)

GO

CREATE TABLE [Unit\_cost]

(

[id] int NOT NULL IDENTITY (1, 1),

[unit\_id] int NOT NULL,

[resource\_id] int NOT NULL,

[cost] int NOT NULL

)

GO

CREATE TABLE [Units]

(

[id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NULL

)

GO

CREATE TABLE [Units\_on\_location]

(

[id] int NOT NULL IDENTITY (1, 1),

[unit\_id] int NOT NULL,

[location\_id] int NOT NULL,

[Count] int NOT NULL

)

GO

CREATE TABLE [User\_account]

(

[user\_id] int NOT NULL IDENTITY (1, 1),

[first\_name] varchar(50) NOT NULL,

[last\_name] varchar(50) NOT NULL,

[nickname] varchar(50) NOT NULL,

[password] varchar(50) NOT NULL,

[gender] varchar(1) NOT NULL,

[user\_role\_id] int NOT NULL,

[email] varchar(50) NOT NULL

)

GO

CREATE TABLE [User\_account\_history]

(

[id] int NOT NULL IDENTITY (1, 1),

[user\_account\_id] int NOT NULL,

[signin\_time] bigint NOT NULL,

[signout\_time] bigint NULL,

[device\_info] varchar(50) NOT NULL,

[ip\_address] varchar(50) NOT NULL

)

GO

CREATE TABLE [User\_opportunity]

(

[op\_id] tinyint NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL

)

GO

CREATE TABLE [User\_permissions]

(

[perm\_id] int NOT NULL IDENTITY (1, 1),

[user\_role\_id] int NOT NULL,

[user\_permissions\_id] tinyint NOT NULL

)

GO

CREATE TABLE [User\_roles]

(

[role\_id] int NOT NULL IDENTITY (1, 1),

[name] varchar(50) NOT NULL

)

GO

/\* Create Primary Keys, Indexes, Uniques, Checks \*/

ALTER TABLE [Clan]

ADD CONSTRAINT [PK\_Clan]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

ALTER TABLE [Clan\_member]

ADD CONSTRAINT [PK\_Clan\_member]

PRIMARY KEY CLUSTERED ([member\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Clan\_member\_Clan]

ON [Clan\_member] ([clan\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Clan\_member\_Clan\_role]

ON [Clan\_member] ([clan\_role\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Clan\_member\_User\_account]

ON [Clan\_member] ([user\_id] ASC)

GO

ALTER TABLE [Clan\_role]

ADD CONSTRAINT [PK\_Clan\_role]

PRIMARY KEY CLUSTERED ([role\_id] ASC)

GO

ALTER TABLE [Group\_members]

ADD CONSTRAINT [PK\_Group\_members]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Group\_members\_Group\_movement]

ON [Group\_members] ([group\_move\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Group\_members\_Units]

ON [Group\_members] ([unit\_id] ASC)

GO

ALTER TABLE [Group\_movement]

ADD CONSTRAINT [PK\_Group\_movement]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Group\_movement\_Location]

ON [Group\_movement] ([location\_from] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Group\_movement\_Location\_02]

ON [Group\_movement] ([location\_to] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Group\_movement\_User\_account]

ON [Group\_movement] ([user\_id] ASC)

GO

ALTER TABLE [Location]

ADD CONSTRAINT [PK\_Location]

PRIMARY KEY CLUSTERED ([location\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Location\_User\_account]

ON [Location] ([user\_id] ASC)

GO

ALTER TABLE [Location\_build]

ADD CONSTRAINT [PK\_Location\_structure]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Location\_structure\_Location]

ON [Location\_build] ([location\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Location\_structure\_Structure]

ON [Location\_build] ([structure\_id] ASC)

GO

ALTER TABLE [Member\_history]

ADD CONSTRAINT [PK\_Member\_history]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Member\_history\_Clan\_member]

ON [Member\_history] ([clan\_member\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Member\_history\_Clan\_role]

ON [Member\_history] ([clan\_role\_id] ASC)

GO

ALTER TABLE [Member\_opportunity]

ADD CONSTRAINT [PK\_Member\_opportunity]

PRIMARY KEY CLUSTERED ([op\_id] ASC)

GO

ALTER TABLE [Necessary\_research]

ADD CONSTRAINT [PK\_prerequisite\_research]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_prerequisite\_research\_Research]

ON [Necessary\_research] ([research\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_prerequisite\_research\_Structure]

ON [Necessary\_research] ([structure\_id] ASC)

GO

ALTER TABLE [Research]

ADD CONSTRAINT [PK\_Research]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

ALTER TABLE [Research\_level]

ADD CONSTRAINT [PK\_Table2]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_level\_User\_account]

ON [Research\_level] ([user\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Table2\_Research]

ON [Research\_level] ([research\_id] ASC)

GO

ALTER TABLE [Research\_requared]

ADD CONSTRAINT [PK\_Research\_requared]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_requared\_Research]

ON [Research\_requared] ([research\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_requared\_Research\_02]

ON [Research\_requared] ([research\_required\_id] ASC)

GO

ALTER TABLE [Research\_unit]

ADD CONSTRAINT [PK\_Research\_unit]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_unit\_Research]

ON [Research\_unit] ([research\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_unit\_Units]

ON [Research\_unit] ([unit\_id] ASC)

GO

ALTER TABLE [Research\_upgrade]

ADD CONSTRAINT [PK\_Research\_formula]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_formula\_Research]

ON [Research\_upgrade] ([research\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Research\_formula\_Resource]

ON [Research\_upgrade] ([resource\_id] ASC)

GO

ALTER TABLE [Resource]

ADD CONSTRAINT [PK\_Resource]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

ALTER TABLE [Resources\_in\_group]

ADD CONSTRAINT [PK\_Resources\_in\_group]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Resources\_in\_group\_Group\_movement]

ON [Resources\_in\_group] ([group\_move\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Resources\_in\_group\_Resource]

ON [Resources\_in\_group] ([resource\_id] ASC)

GO

ALTER TABLE [Resources\_on\_locations]

ADD CONSTRAINT [PK\_Resources\_on\_locations]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Resources\_on\_locations\_Location]

ON [Resources\_on\_locations] ([location\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Resources\_on\_locations\_Resource]

ON [Resources\_on\_locations] ([resource\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Role\_permissions\_Member\_opportunity]

ON [Role\_permissions] ([member\_permissions\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Role\_permissionsf\_Clan\_role]

ON [Role\_permissions] ([member\_role\_id] ASC)

GO

ALTER TABLE [Structure]

ADD CONSTRAINT [PK\_Structure]

PRIMARY KEY CLUSTERED ([str\_id] ASC)

GO

ALTER TABLE [Structure\_required]

ADD CONSTRAINT [PK\_Structure\_required]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Structure\_required\_Structure]

ON [Structure\_required] ([structure\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Structure\_required\_Structure\_02]

ON [Structure\_required] ([structure\_required\_id] ASC)

GO

ALTER TABLE [Structure\_upgrade]

ADD CONSTRAINT [PK\_Structure\_dormula]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Structure\_dormula\_Resource]

ON [Structure\_upgrade] ([resource\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Structure\_dormula\_Structure]

ON [Structure\_upgrade] ([structure\_id] ASC)

GO

ALTER TABLE [Unit\_characteristic]

ADD CONSTRAINT [PK\_Unit\_characteristic]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Unit\_characteristic\_Units]

ON [Unit\_characteristic] ([unit\_id] ASC)

GO

ALTER TABLE [Unit\_cost]

ADD CONSTRAINT [PK\_Table4]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Unit\_cost\_Resource]

ON [Unit\_cost] ([resource\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Unit\_cost\_Units]

ON [Unit\_cost] ([unit\_id] ASC)

GO

ALTER TABLE [Units]

ADD CONSTRAINT [PK\_Units]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

ALTER TABLE [Units\_on\_location]

ADD CONSTRAINT [PK\_Units\_on\_location]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Units\_on\_location\_Location]

ON [Units\_on\_location] ([location\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_Units\_on\_location\_Units]

ON [Units\_on\_location] ([unit\_id] ASC)

GO

ALTER TABLE [User\_account]

ADD CONSTRAINT [PK\_User\_account]

PRIMARY KEY CLUSTERED ([user\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_User\_account\_User\_roles]

ON [User\_account] ([user\_role\_id] ASC)

GO

ALTER TABLE [User\_account\_history]

ADD CONSTRAINT [PK\_User\_account\_history]

PRIMARY KEY CLUSTERED ([id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_User\_account\_history\_User\_account]

ON [User\_account\_history] ([user\_account\_id] ASC)

GO

ALTER TABLE [User\_opportunity]

ADD CONSTRAINT [PK\_User\_opportunity]

PRIMARY KEY CLUSTERED ([op\_id] ASC)

GO

ALTER TABLE [User\_permissions]

ADD CONSTRAINT [PK\_User\_permissions]

PRIMARY KEY CLUSTERED ([perm\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_User\_permissions\_User\_opportunity]

ON [User\_permissions] ([user\_permissions\_id] ASC)

GO

CREATE NONCLUSTERED INDEX [IXFK\_User\_permissions\_User\_roles]

ON [User\_permissions] ([user\_role\_id] ASC)

GO

ALTER TABLE [User\_roles]

ADD CONSTRAINT [PK\_User\_roles]

PRIMARY KEY CLUSTERED ([role\_id] ASC)

GO

/\* Create Foreign Key Constraints \*/

ALTER TABLE [Clan\_member] ADD CONSTRAINT [FK\_Clan\_member\_Clan]

FOREIGN KEY ([clan\_id]) REFERENCES [Clan] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Clan\_member] ADD CONSTRAINT [FK\_Clan\_member\_Clan\_role]

FOREIGN KEY ([clan\_role\_id]) REFERENCES [Clan\_role] ([role\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Clan\_member] ADD CONSTRAINT [FK\_Clan\_member\_User\_account]

FOREIGN KEY ([user\_id]) REFERENCES [User\_account] ([user\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Group\_members] ADD CONSTRAINT [FK\_Group\_members\_Group\_movement]

FOREIGN KEY ([group\_move\_id]) REFERENCES [Group\_movement] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Group\_members] ADD CONSTRAINT [FK\_Group\_members\_Units]

FOREIGN KEY ([unit\_id]) REFERENCES [Units] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Group\_movement] ADD CONSTRAINT [FK\_Group\_movement\_Location]

FOREIGN KEY ([location\_from]) REFERENCES [Location] ([location\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Group\_movement] ADD CONSTRAINT [FK\_Group\_movement\_Location\_02]

FOREIGN KEY ([location\_to]) REFERENCES [Location] ([location\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Group\_movement] ADD CONSTRAINT [FK\_Group\_movement\_User\_account]

FOREIGN KEY ([user\_id]) REFERENCES [User\_account] ([user\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Location\_build] ADD CONSTRAINT [FK\_Location\_structure\_Location]

FOREIGN KEY ([location\_id]) REFERENCES [Location] ([location\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Location\_build] ADD CONSTRAINT [FK\_Location\_structure\_Structure]

FOREIGN KEY ([structure\_id]) REFERENCES [Structure] ([str\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Member\_history] ADD CONSTRAINT [FK\_Member\_history\_Clan\_member]

FOREIGN KEY ([clan\_member\_id]) REFERENCES [Clan\_member] ([member\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Necessary\_research] ADD CONSTRAINT [FK\_prerequisite\_research\_Research]

FOREIGN KEY ([research\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Necessary\_research] ADD CONSTRAINT [FK\_prerequisite\_research\_Structure]

FOREIGN KEY ([structure\_id]) REFERENCES [Structure] ([str\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_level] ADD CONSTRAINT [FK\_Research\_level\_User\_account]

FOREIGN KEY ([user\_id]) REFERENCES [User\_account] ([user\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_level] ADD CONSTRAINT [FK\_Table2\_Research]

FOREIGN KEY ([research\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_requared] ADD CONSTRAINT [FK\_Research\_requared\_Research]

FOREIGN KEY ([research\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_requared] ADD CONSTRAINT [FK\_Research\_requared\_Research\_02]

FOREIGN KEY ([research\_required\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_unit] ADD CONSTRAINT [FK\_Research\_unit\_Research]

FOREIGN KEY ([research\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_unit] ADD CONSTRAINT [FK\_Research\_unit\_Units]

FOREIGN KEY ([unit\_id]) REFERENCES [Units] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_upgrade] ADD CONSTRAINT [FK\_Research\_formula\_Research]

FOREIGN KEY ([research\_id]) REFERENCES [Research] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Research\_upgrade] ADD CONSTRAINT [FK\_Research\_formula\_Resource]

FOREIGN KEY ([resource\_id]) REFERENCES [Resource] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Resources\_in\_group] ADD CONSTRAINT [FK\_Resources\_in\_group\_Group\_movement]

FOREIGN KEY ([group\_move\_id]) REFERENCES [Group\_movement] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Resources\_in\_group] ADD CONSTRAINT [FK\_Resources\_in\_group\_Resource]

FOREIGN KEY ([resource\_id]) REFERENCES [Resource] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Resources\_on\_locations] ADD CONSTRAINT [FK\_Resources\_on\_locations\_Location]

FOREIGN KEY ([location\_id]) REFERENCES [Location] ([location\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Resources\_on\_locations] ADD CONSTRAINT [FK\_Resources\_on\_locations\_Resource]

FOREIGN KEY ([resource\_id]) REFERENCES [Resource] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Role\_permissions] ADD CONSTRAINT [FK\_Role\_permissions\_Member\_opportunity]

FOREIGN KEY ([member\_permissions\_id]) REFERENCES [Member\_opportunity] ([op\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Role\_permissions] ADD CONSTRAINT [FK\_Role\_permissionsf\_Clan\_role]

FOREIGN KEY ([member\_role\_id]) REFERENCES [Clan\_role] ([role\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Structure\_required] ADD CONSTRAINT [FK\_Structure\_required\_Structure]

FOREIGN KEY ([structure\_id]) REFERENCES [Structure] ([str\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Structure\_required] ADD CONSTRAINT [FK\_Structure\_required\_Structure\_02]

FOREIGN KEY ([structure\_required\_id]) REFERENCES [Structure] ([str\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Structure\_upgrade] ADD CONSTRAINT [FK\_Structure\_dormula\_Resource]

FOREIGN KEY ([resource\_id]) REFERENCES [Resource] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Structure\_upgrade] ADD CONSTRAINT [FK\_Structure\_dormula\_Structure]

FOREIGN KEY ([structure\_id]) REFERENCES [Structure] ([str\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Unit\_characteristic] ADD CONSTRAINT [FK\_Unit\_characteristic\_Units]

FOREIGN KEY ([unit\_id]) REFERENCES [Units] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Unit\_cost] ADD CONSTRAINT [FK\_Unit\_cost\_Resource]

FOREIGN KEY ([resource\_id]) REFERENCES [Resource] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Unit\_cost] ADD CONSTRAINT [FK\_Unit\_cost\_Units]

FOREIGN KEY ([unit\_id]) REFERENCES [Units] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Units\_on\_location] ADD CONSTRAINT [FK\_Units\_on\_location\_Location]

FOREIGN KEY ([location\_id]) REFERENCES [Location] ([location\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [Units\_on\_location] ADD CONSTRAINT [FK\_Units\_on\_location\_Units]

FOREIGN KEY ([unit\_id]) REFERENCES [Units] ([id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [User\_account] ADD CONSTRAINT [FK\_User\_account\_User\_roles]

FOREIGN KEY ([user\_role\_id]) REFERENCES [User\_roles] ([role\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [User\_account\_history] ADD CONSTRAINT [FK\_User\_account\_history\_User\_account]

FOREIGN KEY ([user\_account\_id]) REFERENCES [User\_account] ([user\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [User\_permissions] ADD CONSTRAINT [FK\_User\_permissions\_User\_opportunity]

FOREIGN KEY ([user\_permissions\_id]) REFERENCES [User\_opportunity] ([op\_id]) ON DELETE No Action ON UPDATE No Action

GO

ALTER TABLE [User\_permissions] ADD CONSTRAINT [FK\_User\_permissions\_User\_roles]

FOREIGN KEY ([user\_role\_id]) REFERENCES [User\_roles] ([role\_id]) ON DELETE No Action ON UPDATE No Action

GO

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | Наименование | | | | Дополнительные сведения | | |
|  | | | | Текстовые документы | | | |  | | |
|  | | | |  | | | |  | | |
| БГУИР КР 1–40 01 01 104 ПЗ | | | | Пояснительная записка | | | | 47 с. | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | | Графические документы | | | |  | | |
|  | | | |  | | | |  | | |
| ГУИР 951001 104 CП | | | | Инфологическая модель базы данных «База данных стратегической онлайн-игры» | | | | Формат А1 | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  | | | |  | | | |  | | |
|  |  |  |  |  | БГУИР КР 1-40 01 01 104 Д1 | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Изм. | Л. | № докум. | Подп. | Дата | База данных стратегической онлайн-игры |  | | | Лист | Листов |
| Разраб. | | Белых Д.В. |  | 16.12.22 | Т |  |  | 47 | 47 |
| Пров. | | Фадеева. Е.Е |  | 16.12.22 | Кафедра ПОИТ гр. 951001 | | | | |
|  | |  |  |  |
|  | |  |  |  |
|  | |  |  |  |